J Intell Robot Syst (2017) 85:539–552
DOI 10.1007/s10846-016-0388-2

Set-point Control of Mobile Robot with Obstacle Detection and Avoidance Using Navigation Function - Experimental Verification

Wojciech Kowalczyk · Mateusz Przybyla · Krzysztof Kozlowski

Received: 7 March 2016 / Accepted: 24 May 2016 / Published online: 16 July 2016

Abstract This paper presents the results of an experimental verification of mobile robot control algorithm including obstacle detection and avoidance. The controller is based on the navigation potential function that was proposed in work (Urakubo, Nonlinear Dyn. **81**(3), 1475–1487 2015). Conducted experiments considered the task of reaching and stabilization of robot in point. The navigation potential agregates information of robot position and orientation but also the repelling potentials of obstacles. The obstacle detection is performed solely with the use of laser scanner. The experiments show that the method can easily handle environments with one or two obstacles even if they instantly hide or show-up due to the scanner range limits. The experiments also indicate that the utilized control method has a good potential for being used in parallel parking task.

Keywords Mobile robot control · Navigation function · Set point control · Obstacle avoidance

W. Kowalczyk (✉) · M. Przybyla · K. Kozlowski
Chair of Control and Systems Engineering, Faculty of Computing, Poznan University of Technology, Piotrowo 3A, Poznan, Poland
e-mail: wojciech.kowalczyk@put.poznan.pl

1 Introduction

Artificial potential functions (APF) are widely used in robotics since 1986 when they were proposed by Khatib [5]. The idea of repulsive and attractive interactions used to avoid collisions and move to desired position in the original version had some limitations. The main one was possibility of local minima when the APF was not designed carefully.

In the beginning of ’90 Rimon and Koditschek presented concept of the navigation function. At first a sphere world version was introduced [12] that assumes that the obstacles are bounded with spheres in three dimensional space or with circles in planar case. Then the method was expanded to more complex environments [13, 14] and [15]. All these algorithms assumed a point-like robot without constraints.

In 2004 Urakubo [26] expanded the method introducing navigation function that takes into account nonholonomic constraints of the mobile robot. In his method the orientation of the robot can reach desired value just as both position coordinates. Convergence proof was included in the paper.

In the papers [1, 20], and [19] navigation function was used to control multiple mobile robots. Authors of these publications adress the problem of collision avoidance in multiagent robotic systems. In the first of the mentioned papers extension of

the navigation function called multi-robot navigation functions (MRNFs) was applied. Second and third of these papers propose the use of prioritization to solve conflicts in case of concurrent goals of the agents. Examples of navigation function used for collision avoidance in 3D space were presented among others in work [21] and [22]. Authors conducted numerical simulations to show the effectiveness of the method. Work [10] introduces the concept of Lyapunov-like barrier functions which is another kind of potential function. The authors of this work apply this concept to the problem of multi-robot control system. In article [27] nonlinear control for nonholonomic mobile robot tracking target and avoiding collisions is presented. Method takes into account the uncertainty of measurements. The paper includes simulation results illustrating robustness of the proposed design scheme. In [8] simulation results for the group of nonholonomic mobile robots tracking trajectory with leader-follower scheme are presented. Papers [16] and [24] present collision avoidance for multiple robots. The avoidance procedure is activated only close to obstacles. In [17] and [18] measurement uncertainties in addition were taken into account.

Authors of work [6] use a local potential function to control formation of mobile robots in an environment with obstacles. Local artificial potential functions are widely investigated and applied with various robots, including UAVs, however, in this approach local minima may occur. In contrast to the navigation function these methods give no general solution of local minima problem. For the case of navigation function algorithm, there exist a clear methodology of tuning, that guarantees only one global minimum.

In this paper experimental results for the algorithm presented in [25] are shown. In comparison to other approaches it takes into account nonholonomic constraints and guarantees global convergence of the position and orientation coordinates. Despite having these advantages it was not previously tested in practice, especially with on-line obstacle detection. In the authors' previous publication [7] the same control algorithm was investigated, however, in these tests other kind of the mobile platform was used. Moreover, in the previous experiments the location and size of the obstacles were known a priori. In the current research the control utilizes the environment model obtained by the reconstruction of the obstacle parameters on the basis of the data from the laser range finder. In comparison to the previously published results it causes two difficulties: building model of the environment in each algorithm repetition and using these uncertain data to control the motion of the mobile platform. In addition OptiTrack motion capture system was used to assure exact set-point control and collect data for the future off-line analysis.

The paper is organized as follows. Section 2 introduces the model of the robot and the environment. Section 3 presents the control algorithm and short explanation of its parameters. Section 4 describes the setup used for the experiments. In particular Section 4.3 presents the obstacle detection algorithm. Section 5 describes the conducted experiments and includes the results with short comments on each of them. The final section includes concluding remarks.

2 Model of the System

The experiments were conducted on a differentially driven mobile robot with a kinematic model given by the following equation:

$$\dot{\vec{q}} = \vec{B}\vec{u}, \tag{1}$$

where vector $\vec{q}^{\top} \triangleq [x \quad y \quad \theta]$ denotes the pose and x, y, θ are position coordinates and orientation of the robot with respect e. Vector $\vec{u}^{\top} \triangleq \begin{bmatrix} v & \omega \end{bmatrix}$ is the control vector with v denoting linear velocity and ω denoting angular velocity of the platform. The input matrix $\vec{B}$ has the following form:

$$\vec{B} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix}. \tag{2}$$

The control algorithm requires the robot task space to be bounded by a conceivable circle which defines the pool of attraction and its potential. The definition of the attracting potential function [14] is:

$$\beta_0 \triangleq \rho_0^2 - ||\vec{r} - \vec{p}_0||^2, \tag{3}$$

where ρ_0 is the radius of the task space, $\vec{r} = [x \quad y]^{\top}$ is the current position of the robot and $\vec{p}_0$ is the center of the task space. The controller design assumes the obstacles to be circular-shaped objects of radiuses ρ_i ($i = 1, ..., N$, N being the number of obstacles) with

their centers located at positions $\vec{p}_i$. The definition of repelling potential function for i-th obstacle is:

$$\beta_i \triangleq ||\vec{r} - \vec{p}_i||^2 - \rho_i^2. \tag{4}$$

It is clearly seen that the attracting potential function behaves as a negative obstacle, inside which the robot moves.

3 Controller Design

Given an environment with N obstacles and a task of stabilizing the robot in its origin the total navigation potential is defined as:

$$V \triangleq \frac{C}{(C^\kappa + \beta)^{\frac{1}{\kappa}}}, \tag{5}$$

where κ is a positive, constant design parameter and

$$C \triangleq ||\vec{r}||^2 + \theta^2 \frac{k_w}{k_w + ||\vec{r}||^2}. \tag{6}$$

Symbol k_w in Eq. 6 denotes a positive, constant design parameter that allows to tune the influence of the orientation term on the navigation function depending on the euclidean distance to the goal. The aggregation of repelling potentials happens in term β which is defined as:

$$\beta \triangleq \prod_{i=0}^{N} \beta_i. \tag{7}$$

One must note that the iteration starts from zero, which means the inclusion of task space boundary potential.

The control algorithm proposed in work [25] is defined as:

$$\vec{u} \triangleq -\left\{ a \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + b \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \right\} \vec{B}^\top \nabla V, \tag{8}$$

where a is a positive, constant design parameter and

$$b \triangleq -\bar{b} \frac{\vec{L}^\top \nabla V}{h(g)}. \tag{9}$$

Symbol $\bar{b}$ in Eq. 9 denotes another positive, constant design parameter and $\vec{L}^\top \triangleq [\sin\theta \quad -\cos\theta \quad 0]$. Function $h(g)$ is defined as:

$$h(g) \triangleq g^2 + \epsilon\sqrt{g}, \tag{10}$$

where $g \triangleq ||\vec{B}^\top \nabla V||$ and ϵ is a small positive constant. Finally, ∇V denotes the gradient of the navigation function with respect to variables x, y and θ. Regardless of number of obstacles, the gradient can be obtained in analytical form as:

$$\nabla V = \frac{\nabla C (C^\kappa + \beta)^{\frac{1}{\kappa}} - \frac{C}{\kappa}(C^\kappa + \beta)^{(\frac{1}{\kappa}-1)}(\kappa C^{\kappa-1}\nabla C + \nabla\beta)}{(C^\kappa + \beta)^{(\frac{2}{\kappa})}}, \tag{11}$$

where

$$\nabla C = \begin{bmatrix} \frac{\partial C}{\partial x} \\ \frac{\partial C}{\partial y} \\ \frac{\partial C}{\partial \theta} \end{bmatrix} = \begin{bmatrix} 2x(1 - \frac{k_w\theta^2}{(k_w + ||\vec{r}||^2)^2}) \\ 2y(1 - \frac{k_w\theta^2}{(k_w + ||\vec{r}||^2)^2}) \\ 2\theta \frac{k_w}{k_w + ||\vec{r}||^2} \end{bmatrix} \tag{12}$$

and

$$\nabla\beta = \sum_{i=0}^{N} \left\{ \frac{\partial \beta_i}{\partial \vec{q}} \prod_{j=0, j\neq i}^{N} \beta_j \right\}. \tag{13}$$

As noted in [14] all the undesired local minima of navigation function (5) disappear as the parameter κ increases. An algorithm for automatically tuning analytic navigation functions for sphere worlds was presented in [3]. The tuning parameter must satisfy a lower bound to ensure convergence to the desired value. In this paper navigation functions have been manually tuned to assure convergence. For the sufficiently high value of the κ parameter navigation function (5) has a critical point associated with each isolated obstacle, the saddle point. V has no other critical point other than these points. Saddle points are unstable equilibrium points. In [25] special control procedure for saddle point avoidance is described. It uses time varying function to push the robot away from the unstable equilibrium point.

4 Experimental Setup

The preparations for the experiments were conducted using simulated environment and robot. The simulations provided a set of parameters that resulted in reasonable behavior of the robot. The experiments introduced few new factors that imposed a need for further retuning. The following section describes the details of the experimental test-bed.

Fig. 1 A picture of a mobile robot used to conduct experiments

4.1 The Hardware

The experiments were performed with the use of MTracker - a small, two-wheeled mobile robot designed and constructed at Chair of Control and Systems Engineering (Fig. 1).

The MTracker is equipped with a single board computer (SBC) with one-core processor (Intel Atom 1.2 GHz) and a low-level controller based on a digital signal processor. A Hokuyo laser scanner is mounted on top of the robot body. The scanner angular and radial range is 240° and 5.6 m, respectively. The average scan rate given by the scanner is 10 Hz.

The robot body is cylinder-shaped with the diameter (wheel base) of 14.5 cm. The wheels diameter is approximately 5 cm. The robot is powered with 3S2P, 3.7 Ah LiIo battery. With the battery mounted, the mass of the robot is 1.47 kg. The top of the robot body is covered with 5 infra-red reflecting markers (one on top of the laser scanner) used for external localization. The localization is performed with the use of ten active, infrared cameras which are part the OptiTrack system.

The SBC on the robot communicates with a personal computer (PC) via Wi-Fi. The average communication delay is 8 ms. The main task of the PC is to provide the user with a graphical user interface (GUI, Fig. 2) but also to collect the data obtained from OptiTrack system and pass it over to the robot.

4.2 The Software

Both the SBC and the PC work on Ubuntu operating system (versions 12.04 and 14.04, respectively). Furthermore, both systems include ROS system (version Hydromedusa on SBC and version Indigo Igloo on PC). The overall system was highly distributed into so-called nodes - isolated processes of ROS. A simplified graph of the system is presented in Fig. 3.

Some of the nodes depicted in Fig. 3 work in an asynchronous (reactive) manner (e.g. Controls Scaling or MTracker). This means that they perform their task immediately after receiving new input data. Some of the nodes work in a synchronous manner (e.g. Obstacle Controller, State Estimator). This means that they perform their task in a cyclical way. The rate of these cycles was set to 100 Hz (ROS does not provide real time operations). In the graph, one can notice a State Estimator node. While conducting experiments this node only copied the OptiTrack data. This was reasoned by the high precision of the external localization method.

The Controls Scaling node scales down control signal $\vec{u}$ when at least one of the wheels exceeds assumed limitation. The scaled control signal $\vec{u}_s$ is calculated as follows:

$$\vec{u}_s = s\vec{u}, \tag{14}$$

where

$$s = \begin{cases} \frac{\omega_{max}}{\omega_o} & \text{if} \quad \omega_o > \omega_{max} \\ 1 & \text{otherwise} \end{cases}, \tag{15}$$

and

$$\omega_o = \max\{|\omega_l|, |\omega_r|\}, \tag{16}$$

where ω_l, ω_r denote left and right wheel angular velocity. Symbol ω_{max} denotes the predefined maximal allowed angular velocity for each wheel. This scaling algorithm preserves the direction of the control vector $\vec{u}$.

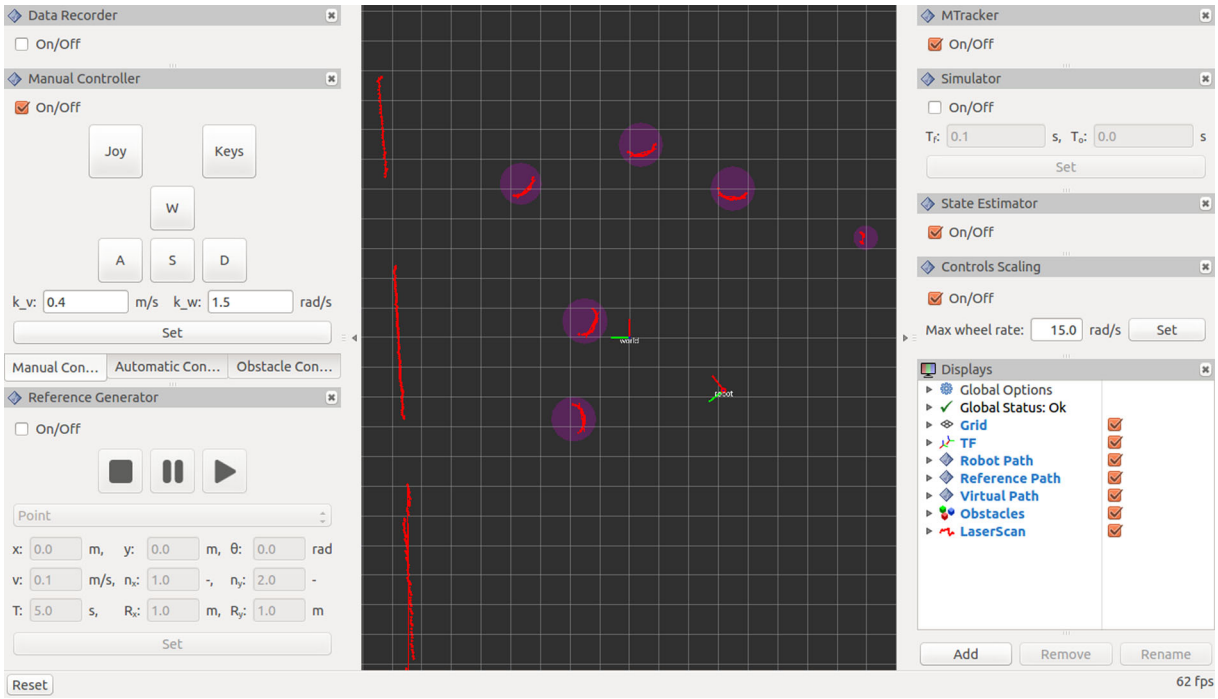


Fig. 2 A screenshot of a graphical user interface with an environment containing 5 obstacles

4.3 The Obstacle Detection

An important part of the system is the obstacle detecting node. It does not solve the obstacle tracking problem, so the obstacles are dynamically recalculated from scan to scan. This introduces a number of phenomena in the potential-based control algorithms. The obstacles are provided in two basic forms: line-shaped and circular-shaped. The control algorithm uses only the latter ones. For the sake of simplicity, the robot workspace has been virtually restricted so that only the essential obstacles were taken into consideration by the algorithm. Still, however, the limitation of scanner angular range caused discontinuous effects while rotating and 'discovering' new obstacles. The obstacle detecting node exploits data given by external localization in order to transform obstacles from local to global coordinate frame. Below we present the steps of obstacle detection algorithm.

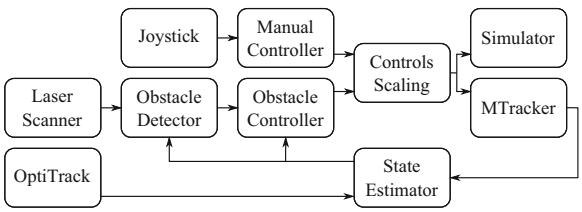


Fig. 3 A graph representation of the system

1. Range data is copied into a buffer of 2D points. While copying, the artificial range restrictions are taken into consideration.
2. Point data is grouped into sets by evaluating a point-to-point distance criterion (in the criterion, the threshold distance grows with the range).
3. Each point set is recursively turned into set of segments with the use of iterative end point fit method [2, 11] (a method similar to split-and-merge algorithm [9]). The segments are reconstructed from point data with the use of total least squares method, where the model of the leading line is represented with general equation: $Ax + By + C = 0$. The vector and matrix algebra was performed with the use of Armadillo C++ library [23]. At last, both first and last point of the set are projected onto the leading line. These two projected points form a segment.
4. An iterative search on total set of segments merges those of them that possibly comprise a

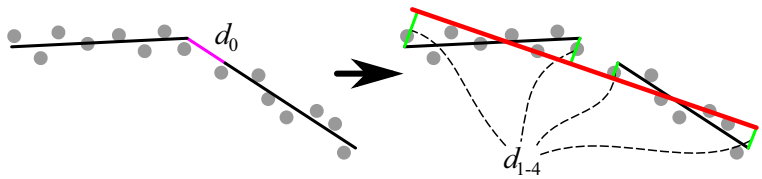


Fig. 4 A graphical visualisation of two segments merge condition. If distance d_0 is smaller than assumed threshold, a new potential segment is created (*red line*). If distances d_{1-4} are also small enough, the new segment replaces the previous two

single object. The criterion evaluated here is composed of two equally important parts: the distance between the neighboring end points of two considered segments must be lower than assumed threshold, if so, a new segment is reconstructed from sum of both point sets and distances between all of the extreme points of both segments must be lower than assumed threshold (Fig. 4). This ends up the segments detection.

5. The detection of circular obstacles is based on the total segments set from the previous step. If a segment length is below assumed threshold, create an equilateral triangle such that the segment becomes its base, and its height points outwards (from the robot). Create a circle circumscribed on this triangle and enlarge its radius by some constant margin. This methodology works more stable than exact circle approximation [4] because not always the obstacles are circular.
6. An iterative search on the total set of circles merges small circles that are close to each other and that do not exceed allowable radius after merge.

In order to show effectiveness of this method, various experiments were conducted. During first experiments set the robot was standing still in the origin of global coordinate frame and few yellow, cylinder-shaped obstacles of radius 13.5 cm have been placed in its workspace. The number of range points constituing detected obstacle decreases with an inverse square of distance and hence the quality of detected parameters declines. Table 1 gathers statistical data (mean values and variances of position and radius) of ten obstacles randomly placed in the workspace. The values of detected radiuses are the true detected values. During control experiments they were enlarged by a margin of 5 cm.

The second experiments set consisted in detecting two non-moving obstacles while the robot was in motion. The robot was moving toward the obstacles with predefined forward velocity and negligible angular rate. Table 2 gathers similar statistical data obtained from the experiments. One can note that the variance of position has largely grown. This is explained by the fact that the central points of detected obstacles do not coincide with the real

Table 1 Statistical data of statically detected obstacles

No.	dist. [m]	$\bar{x}$ [m]	σ_x^2 [mm^2]	$\bar{y}$ [m]	σ_y^2 [mm^2]	$\bar{r}$ [m]	σ_r^2 [mm^2]
1	0.4171	0.4171	1.1907	−0.0087	0.7693	0.1368	1.1001
2	0.7186	0.6020	1.0339	−0.3925	1.8257	0.1467	1.9914
3	0.7599	0.6494	1.2612	0.3946	0.7185	0.1444	0.5515
4	0.7943	0.7939	1.8326	−0.0242	0.6047	0.1425	0.4934
5	1.3306	1.3300	1.8270	−0.0403	0.3874	0.1401	0.15632
6	1.4386	1.3282	2.1739	0.5528	6.1412	0.1399	3.609
7	1.4923	1.3600	1.7138	−0.6142	2.9745	0.1411	0.1455
8	1.9014	1.8984	9.2127	−0.1061	6.6801	0.1381	11.824
9	2.2044	2.1224	3.8768	−0.5957	9.5848	0.1376	9.5685
10	2.2663	2.1869	2.3844	0.5945	5.0300	0.1417	4.7724

Table 2 Statistical data of dynamically detected obstacles

No.	v [m/s]	$\bar{x}$ [m]	σ_x^2 [mm^2]	$\bar{y}$ [m]	σ_y^2 [mm^2]	$\bar{r}$ [m]	σ_r^2 [mm^2]
1	0.125	2.1342	41.774	−0.5745	86.189	0.1395	11.705
2	0.125	2.1923	21.227	0.6035	29.122	0.1409	15.160
3	0.25	2.1434	99.516	−0.5724	163.83	0.1404	12.376
4	0.25	2.1983	46.161	0.6046	77.529	0.1408	15.683
5	0.375	2.1549	119.93	−0.5679	157.67	0.1409	9.3442
6	0.375	2.2077	103.95	0.6079	99.483	0.1417	17.367

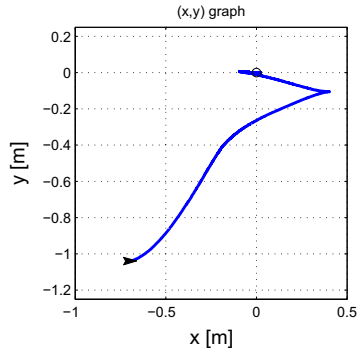
obstacles central points and during the motion the robot 'sees' them from different angles, hence the points wander.

5 Experimental Results

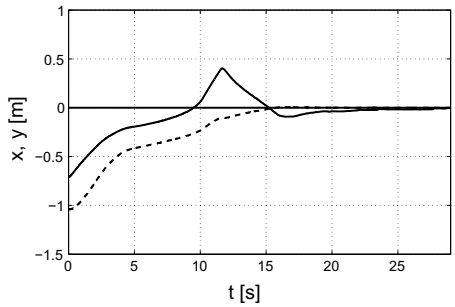
In all of the presented experiments desired position and orientation was set to zero ($\vec{q}_{\mathrm{d}}^{\top} = [0 \quad 0 \quad 0]$). During experiments the positions of obstacles were detected on-line using laser scanner. Algorithm constants used during the experiments were as follows: $a = 0.5$, $\bar{b} = 2.5$, $\epsilon = 10^{-4}$, $k_w = 0.1$ and $\kappa = 3$. These values were obtained during preliminary experiments. The values of a and b coefficients affect the rate of convergence of positions coordinates. They were tuned to get expected time of task execution and avoid disturbances (larger values of the a and b produced larger $\vec{u}$ that led the wheel velocity controller switching between positive and negative velocity limit). Coefficient ϵ were tuned to reduce the chattering in the neighborhood of the desired position. It is worth to note that the value of this coefficient did not cause chattering in wide range during simulation tests (e.g. authors of [25] used value $\epsilon = 10^{-6}$ that gave no acceptable results in tests with real robot). The value of k_w coefficient allows to tune the ratio of the orientation to position convergence rate. Inadequate values caused that the orientation converge to desired value too fast or too slow in comparison to position coordinates. The κ coefficient were used to form the velocity profile along the robots path. By increasing this value local minima of the potential function are flattened and eventually removed. Usually κ must be increased if narrow passages between obstacles exist in the environment. The radius of the task space was set to 5 m. The angular velocities of the wheels were restricted to maximum value of $w_{\max} = 10$ rad/s. This value was selected referring to physical properties of the MTracker robot. Such value provided that there were no longitudinal slides during task execution. In subsequent experiments the algorithm was investigated for increasing complexity of the environment.

In the first experiment the robot moved in the free space. Initial configuration was as follows: $\vec{q}^{\top}(0) = [-0.71 \quad -1.04 \quad 0.015]$. During the maneuver the robot changed the direction of motion twice (Fig. 5a). The non-trivial path of the robot is the result of the influence of orientation component of the navigation function. The robot reached the neighborhood of the desired position after about 20 seconds (Fig. 5b). Desired orientation converged at the same time (Fig. 5c). Figure 5d and e show the linear and angular velocities of the platform, respectively. The dashed line represents the original control signal computed with Eq. 8, the solid one represents scaled control signals. In Fig. 5g wheels velocities are presented. In Fig. 5f time graph of the navigation function computed along motion path is shown. The influence of the potential function associated with the boundary of the task space is not significant as the robot stays far from the boundary during the task execution.

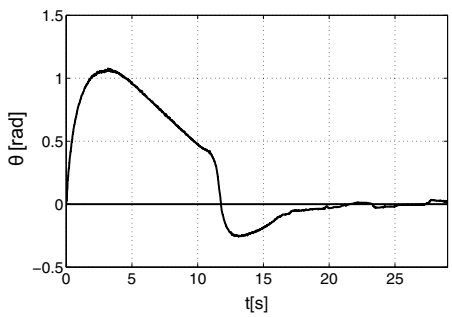
In Fig. 6 results for the experiment with single obstacle are presented. Robot initial coordinates were as follows: $\vec{q}^{\top}(0) = [-1.76 \quad -0.30 \quad -0.81]$. During the task execution robot avoided the obstacle in a smooth fashion (Fig. 6a), reaching desired position after about 25 seconds (Fig. 6b and c). Figure 6d and e show the linear and angular velocities of the platform. Again, dashed lines represent control signals without scaling. The solid lines represent scaled



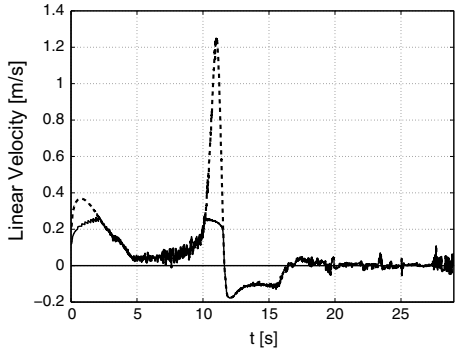
(a) (x,y)-paths



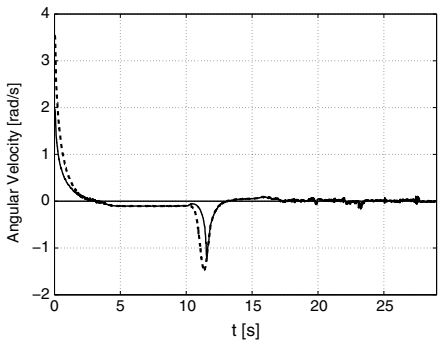
(b) Position components: x - solid line, y - dashed line



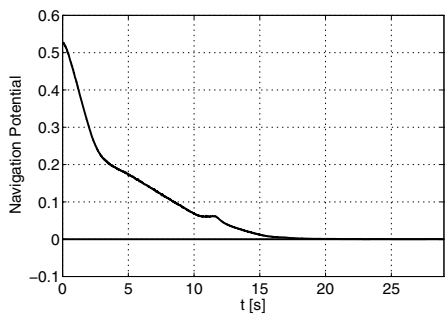
(c) Orientation



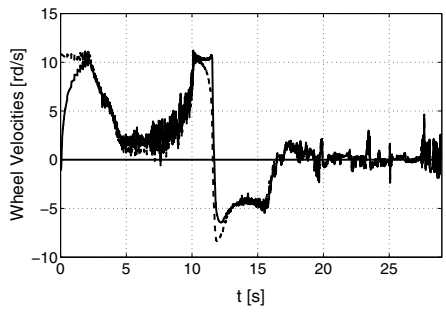
(d) Linear velocity



(e) Angular velocity

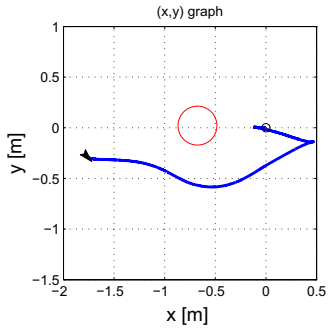


(f) Artificial potential

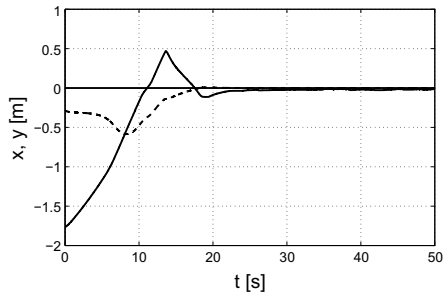


(g) Wheel velocities

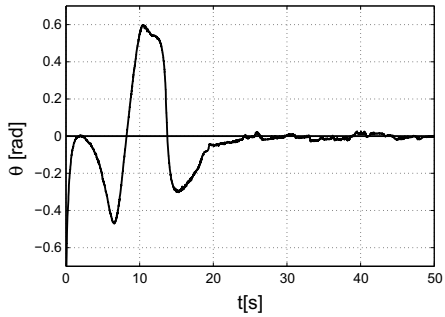
Fig. 5 Experiment 1: motion in the free space



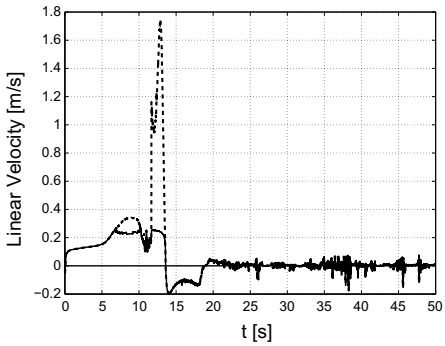
(a) (x,y)-paths



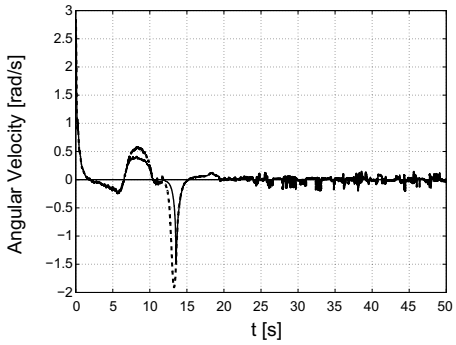
(b) Position components: x - solid line, y - dashed line



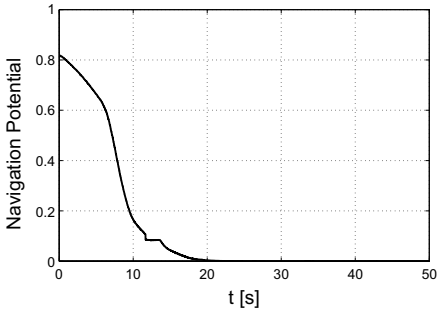
(c) Orientation



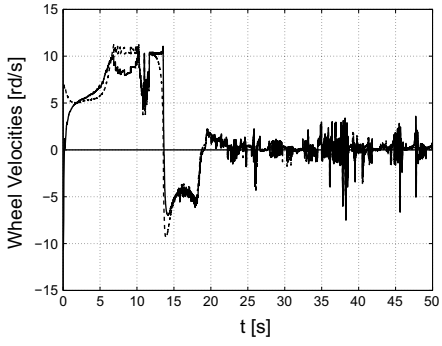
(d) Linear velocity



(e) Angular velocity

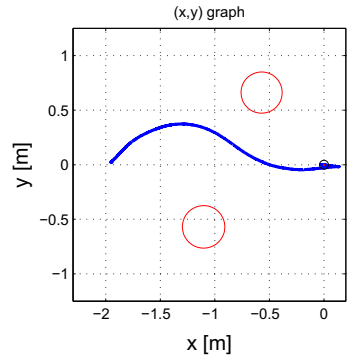


(f) Artificial potential

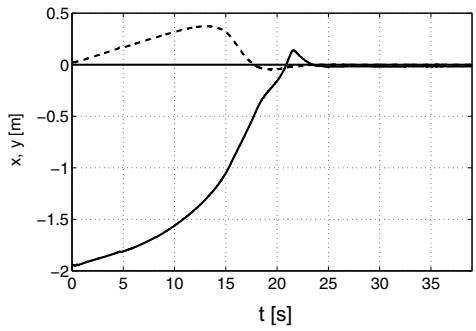


(g) Wheel velocities

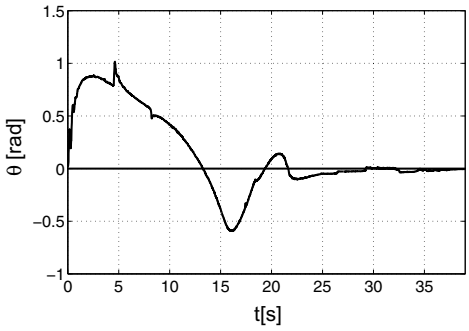
Fig. 6 Experiment 2: space with one obstacle



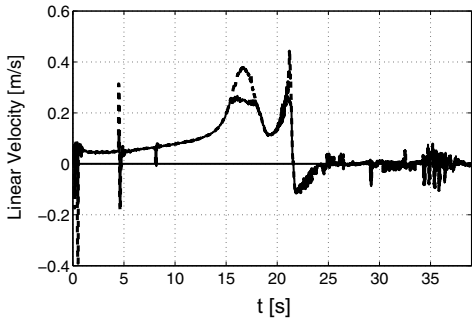
(a) (x,y)-paths



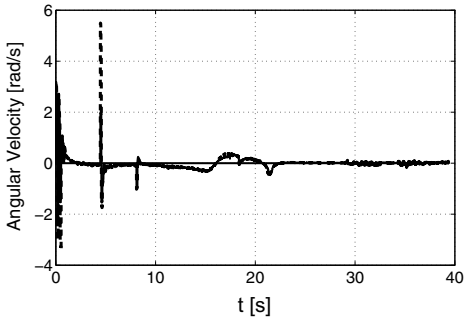
(b) Position components: x - solid line, y - dashed line



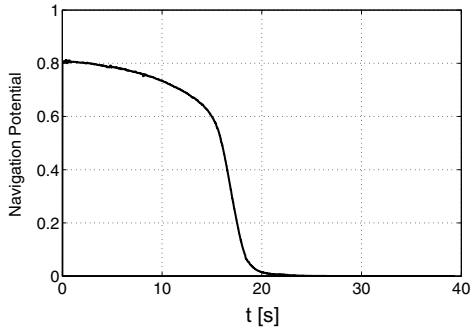
(c) Orientation



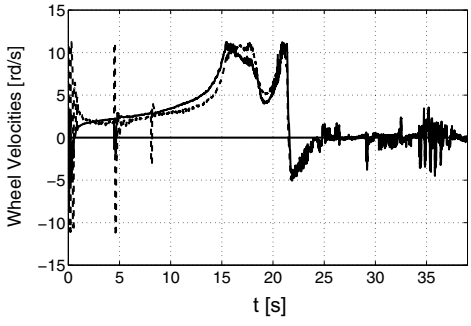
(d) Linear velocity



(e) Angular velocity



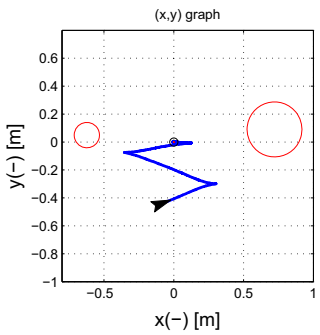
(f) Artificial potential



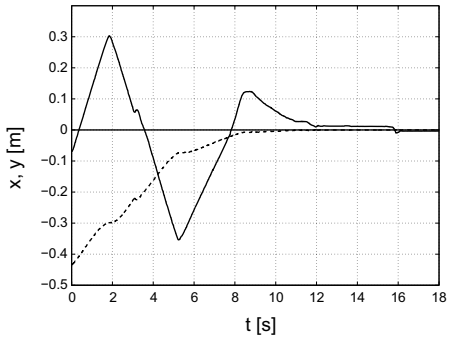
(g) Wheel velocities

Fig. 7 Experiment 3: space with two obstacles

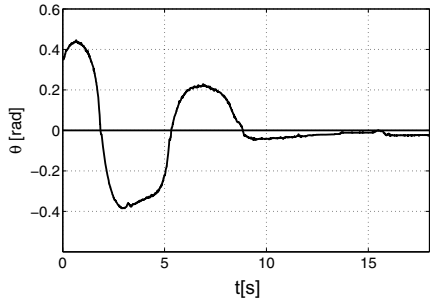
Fig. 8 Experiment 4: parallel parking maneuver



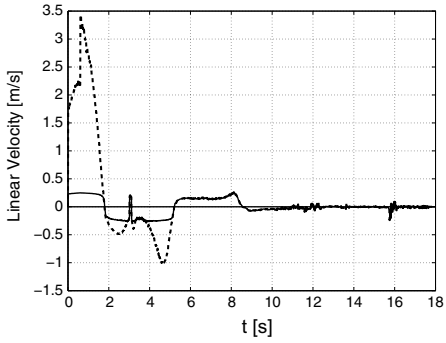
(a) (x,y)-paths



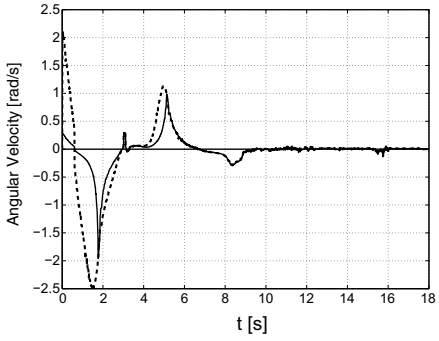
(b) Position components: x - solid line, y - dashed line



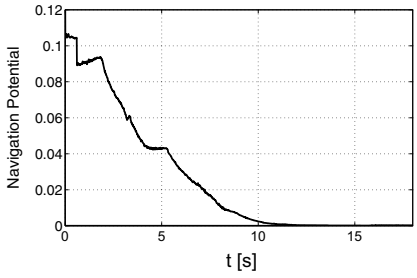
(c) Orientation



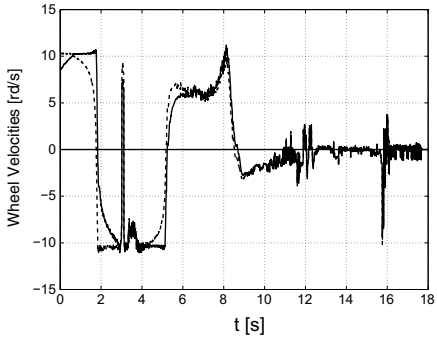
(d) Linear velocity



(e) Angular velocity



(f) Artificial potential



(g) Wheel velocities

control signals. In Fig. 6g wheels velocities signals are presented. Figure 6f represents the time graph of the potential function. It is worth noting the discontinuity at about 12 second. It is the result of obstacle disappearing when the robot moved behind it. Although the linear velocity has a drastic impulse about that time, the scaling procedure saved the robot from such reaction and thus the motion remained smooth. The influence of the environment on the robots motion is significant as the obstacle is located between the initial and final positions of the robot. Initially robot moves toward the goal but as it approaches the obstacle the bypass maneuver is executed.

Figure 7a shows the robot path for the environment with two obstacles. Robot started the motion in the position $\vec{q}^{\top}(0) = [-1.95 \quad 0.02 \quad 0.02]$. Desired position and orientation are reached after about 27 seconds (Fig. 7b and c). In Fig. 7d and f linear and angular velocities of the platform are presented. Grey lines represent signal without scaling and the solid lines are scaled control signals. In Fig. 7g wheel angular velocities are presented. Figure 7f shows the navigation function graph. In this scenario two obstacles formed the shape of the resulting robots path. It keeps far from the obstacle during the whole task execution. In the initial part of the path the tendency to push the robot into empty areas can be observed.

The results of the last experiment are shown in Fig. 8. In this scenario robot executes parallel parking maneuver. In Fig. 8a the path of the robot is shown. In Fig. 8b and c position and orientation coordinates are presented. The robot reaches the neighbourhood of the desired value after about 14 seconds. In Fig. 8d and e control signals for the platform are presented. In Fig. 8g wheel controls are presented. The potential function is shown in Fig. 8f. Due to the limited angular range of the laser range finder temporary growth of the potential is observed during initial motion. When the robot rotates left at the beginning the left obstacle falls within the scope of detection. In this test the robot operates close to the obstacles, hence their influence on the control signals is substantial. It results in three changes of the motion direction but finally desired position between two obstacles is reached. Note that the desired orientation causes inconvenience as one obstacle is in the front of the robot while the second is at its back during the final part of the task execution.

6 Conclusion

In this paper experimental verification of the nonholonomic mobile robot control and collision avoidance algorithm was presented. Both position errors and orientation error are quickly reduced to near zero values. Future research will focus on the more complex environments, including star shaped obstacles and tree of stars obstacles. Authors plan to expand the algorithm to adopt it to multiple nonholonomic robot control and dynamic environment. The saddle point avoidance procedure described in [25] will be tested in the near future.

References

1. Dimarogonasa, D.V., Loizoua, S.G., Kyriakopoulos, K.J., Zavlanosb, M.M.: A feedback stabilization and collision avoidance scheme for multiple independent non-point agents. Automatica **42**(2), 229–243 (2005)
2. Douglas, D., Peucker, T.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. In: The Canadian Cartographer, pp. 112–122 (1973)
3. Filippidis, I., Kyriakopoulos, K.J.: Adjustable navigation functions for unknown sphere worlds. In: IEEE Conference on Decision and Control and European Control Conference (CDC-ECC), pp. 4276–4281 (2011)
4. Kasa, I.: A circle fitting procedure and its error analysis. IEEE Trans. Instrumen. Measur., 8–14 (1976)
5. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. Int. J. Robot. Res. **5**(1), 90–98 (1986)
6. Kowalczyk, W., Kozlowski, K., Tar, J.K.: Trajectory tracking for multiple unicycles in the environment with obstacles. In: International Conference on Robotics in Alpe-Adria-Danube Region (RAAD), pp. 451–456 (2010)
7. Kowalczyk, W., Przybyla, M., Kozlowski, K.: Control of a mobile robot and collision avoidance using navigation

function-experimental verification. In: 2015 10th International Workshop on Robot Motion and Control (RoMoCo), pp. 148–152
8. Loria, A., Dasdemir, J., Alvarez Jarquin, N.: Leader follower formation and tracking control of mobile robots along straight paths. IEEE Trans Control Syst Technol **24**(2), 727–732 (2016)
9. Nguyen, V., Martinelli, A., Tomatis, N., Siegwart, R.: A comparison of line extraction algorithms using 2D laser rangefinder for indoor mobile robotics. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1929–1934 (2005)
10. Panagou, D., Stipanovic, D.M., Voulgaris, P.G.: Distributed coordination control for multi-robot networks using Lyapunov-like barrier functions. IEEE Trans. Autom. Control **61**(3), 617–632 (2016)
11. Ramer, U.: An iterative procedure for the polygonal approximation of plane curves. In: Computer Graphics and Image Processing, pp. 244–256 (1972)
12. Rimon, E., Koditschek, D.E.: Exact robot navigation using cost functions: the case of distinct spherical boundaries. IEEE Int. Conf. Robot. Autom. **3**, 1791–1796 (1988)
13. Rimon, E., Koditschek, D.E.: The construction of analytic diffeomorphisms for exact robot navigation on star worlds. Trans. Amer. Math. Soc. **327**, 71–116 (1991)
14. Rimon, E., Koditschek, D.: Exact robot navigation using artificial potential functions. IEEE Trans. Robot. Autom. **8**(5), 501–518 (1992)
15. Rimon, E., Koditschek, D.E.: Exact robot navigation using artificial potential fields. IEEE Trans. Robot. Autom. **8**(5), 501–518 (1992)
16. Rodriguez-Seda, E.J., Tang, C., Spong, M.W., Stipanovic, D.M.: Trajectory tracking with collision avoidance for nonholonomic vehicles with acceleration constraints and limited sensing. In: The International Journal of Robotics Research, pp. 1–34. USA (2014)
17. Rodriguez-Seda, E.J., Stipanovic, D.M., Spong, M.W.: Collision avoidance control with sensing uncertainties. In: Proc. Am. Control Conf., pp. 3363–3368. San Francisco (2011)
18. Rodriguez-Seda, E.J., Stipanovic, D.M., Spong, M.W.: Lyapunov-based cooperative avoidance control for multiple Lagrangian systems with bounded sensing uncertainties. In: Proc. IEEE Conf. Decision Control, pp. 4207–4213. Orlando (2011)
19. Roussos, G., Kyriakopoulos, K.J.: Completely decentralised navigation of multiple unicycle agents with prioritisation and fault tolerance. In: IEEE Conference on Decision and Control (CDC), pp. 1372–1377 (2010)
20. Roussos, G., Kyriakopoulos, K.J.: Decentralized and prioritized navigation and collision avoidance for multiple mobile robots. Distrib. Autonom. Robot. Syst. Springer Tracts Adv. Robot. **83**, 189–202 (2013)
21. Roussos, G., Kyriakopoulos, K.: Decentralized navigation and conflict avoidance for aircraft in 3-D space. IEEE Trans. Control Syst. Technol. **20**(6), 1622–1629 (2012). doi:10.1109/TCST.2011.2167974.
22. Roussos, G., Dimarogonas, D., Kyriakopoulos, K.: 3D navigation and collision avoidance for nonholonomic aircraft-like vehicles. Int. J. Adaptive Control Signal Process. Special Issue: Air Traff. Manag. Challenges Opportun. Adv. Control **24**(10), 900–920 (2010)
23. Sanderson, A.C.: An open source C++ linear algebra library for fast prototyping and computationally intensive experiments. Technical Report, NICTA (2010)
24. Stipanovic, D.M., Hokayem, P.F., Spong, M.W., Siljak, D.: Cooperative avoidance control for multiagent systems. J. Dyn. Syst. Meas. Control **129**, 699–707 (2007)
25. Urakubo, T.: Feedback stabilization of a nonholonomic system with potential fields: application to a two-wheeled mobile robot among obstacles. Nonlinear Dyn. **81**(3), 1475–1487 (2015)
26. Urakubo, T., Okuma, K., Tada, Y.: Feedback control of a two wheeled mobile robot with obstacle avoidance using potential functions. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) **3**, 2428–2433 (2004)
27. Yang, H., Fan, X., Shi, P., Hua, C.: Nonlinear control for tracking and obstacle avoidance of a wheeled mobile robot with nonholonomic constraint. IEEE Trans. Control Syst. Technol. **24**(2), 741–746 (2016)

Wojciech Kowalczyk received the M.Sc. degree in robotics from Poznan University of Technology (PUT), Poland in 1998. In 1999-2000 he was working as electronics design engineer and software developer (programming databases and communication applications for retail chains). Since 2000 he works at Poznan University of Technology where he received the Ph.D. degree in robotics in 2008. His scientific interests focus on the multi-agent systems, mainly on their application in the control of multiple mobile robots, collision avoidance and cooperative behaviours of non-holonomic robots at both kinematics and dynamics level. He was involved in medical research projects conducted at PUT. He teaches multi-agent systems, smart systems, robotics, object oriented programming, electrical engineering and electronics.

Mateusz Przybyla received his Eng. degree in robotics in 2008 and M.Sc. degree in robotics and management in 2010 from Poznan University of Technology (PUT) in Poland. His master thesis concerned 2D SLAM based on laser scanner data. After graduating in 2010 he became a doctorate student at PUT. His main scientific focus was put on autonomous mobile robots, robust control systems and perception systems. He was involved in several research projects as a programmer. He teaches fundamentals of autonomous systems and basics of control systems. He also gives courses on ROS for students associated with a local scientific club.

Professor K. Kozlowski received the M.Sc. degree in electrical engineering from Poznan University of Technology (PUT), Poland, and the Ph.D. degree in control engineering from PUT in 1979, where he currently holds full professor position in robotics and automation. In 1993, he was a Fulbright scholar with Jet Propulsion Laboratory, Pasadena, USA. He founded and serves as a chairman of a new Institute of Control and Systems Engineering established on January 1, 2002 at Poznan University of Technology.

He teaches and conducts research in the area of modeling and control of industrial and mobile robots. His research interests include multi-agent systems, identification and various robotics applications. His research publications include more than 140 conference papers and more than 50 papers published in national and international journals. He is an author of the book titled Modelling and Identification in Robotics (Springer-Verlag, 1998).

He was an Associate Editor for the IEEE Transactions on Control Systems Technology (1999-2008), for the IEEE Robotics and Automation Magazine (1998-2001) and for the Journal of Intelligent and Robotic Systems (2004-2010). He is an Associate Editor for the International Journal of Applied Mathematics and Computer Science since 1999 till now and for the IEEE Conference Editorial Board, Conference on Decision and Control (CDC) and American Control Conference (ACC) since 1999 till now. He is actively working for IEEE Robotics and Automation Society being a member of The Administration Committee (2001-2002 and 2004-2005) and IEEE Control Systems Society being a member of the Board of Governors (2003-2004). He supports IEEE Robotics and Automation Polish Chapter in the years 2000-2008 and currently 2014-2017. In 2001 this Chapter was awarded the prize “2001 Best Chapter of the Year”.